IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

PATENT APPLICATION
FOR:

METHOD OF INITIALIZING AND USING A SECURITY ASSOCIATION FOR MIDDLEWARE BASED ON PHYSICAL PROXIMITY

INVENTORS:

Sampo SOVIO,
Jan-Erik EKBERG, and
Philip GINZBOORG

Morgan & Finnegan, L.L.P.
345 Park Avenue
New York, New York 10154-0053
(212) 758-4800
(212) 751-6849 (Facsimile)
www.MorganFinnegan.com

Attorneys for Applicant

METHOD OF INITIALIZING AND USING A SECURITY ASSOCIATION FOR MIDDLEWARE BASED ON PHYSICAL PROXIMITY

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application for letters patent is related to and incorporates by reference United States patent application serial number 10/284,135, titled "DEVICE DETECTION AND SERVICE DISCOVERY SYSTEM AND METHOD FOR A MOBILE AD HOC COMMUNICATIONS NETWORK", and filed in the United States Patent and Trademark Office on October 31, 2002. This application for letters patent is also related to and incorporates by reference United States continuation-in-part patent application serial number 10/662,407, titled "DEVICE DETECTION AND SERVICE DISCOVERY SYSTEM AND METHOD FOR A MOBILE AD HOC COMMUNICATIONS NETWORK", and filed in the United States Patent and Trademark Office on September 16, 2003. This application for letters patent is also related to and incorporates by reference United States patent application serial number 10/662,470, titled "MECHANISM FOR IMPROVING CONNECTION CONTROL IN PEER-TO-PEER AD-HOC NETWORKS", and filed in the United States Patent and Trademark Office on September 16, 2003. This application for letters patent is also related to and incorporates by reference United States patent application serial number 10/662,469, titled "APPLICATION CONTROL IN PEER-TO-PEER AD-HOC COMMUNICATION NETWORKS", and filed in the United States Patent and Trademark Office on September 16, 2003. The assignee is the same in this application and the related patent applications.

FIELD OF THE INVENTION

[0002] The present invention relates, in general, to communication between devices connected to a wireless communication network. In particular, the present invention is a system and method for launching and controlling secure and non-secure application programs in wireless devices in a mobile ad-hoc communications network.

BACKGROUND OF THE INVENTION

[0003] Short-range wireless systems have a range of less than one hundred meters, but may connect to the Internet to provide communication over longer distances. Short-range wireless systems include, but are not limited to, a wireless personal area network (PAN) and a wireless local area network (LAN). A wireless PAN uses low-cost, low-power wireless devices that have a typical range of ten meters. An example of a wireless PAN technology is the Bluetooth Standard. The Bluetooth Standard operates in the 2.4 GHz Industrial, Scientific, and Medical (ISM) band and provides a peak air-link speed of one Mbps and a power consumption low enough for use in personal, portable electronics such as a personal digital assistance or mobile phone. A introduction to Bluetooth applications is in *Bluetooth Application Developer's Guide: The Short Range Interconnect Solution, Chapter 1*, Syngress Publishing, Inc., 2002. Another example of a wireless PAN technology is a standard for transmitting data via infrared light waves developed by the Infrared Data Association (IrDA), a group of device manufacturers. IrDA ports enable computers, such as a laptop, or devices, such as a printer, to transfer data from one device to another without any cables. IrDA ports support roughly the same transmission rates as traditional parallel ports and the only restrictions on their use is that the two devices must be proximately located (i.e., within a few feet of each other) and have a clear line of sight. A

wireless LAN is more costly than a wireless PAN, but has a longer range. An example of a wireless LAN technology is the IEEE 802.11 Wireless LAN Standard and the HIPERLAN Standard. The HIPERLAN Standard operates in the 5 GHz Unlicensed-National Information Infrastructure (U-NII) band and provides a peak air-link speed between ten and one hundred Mbps.

[0004] An ad-hoc network is a short-range wireless system comprising an arbitrary collection of wireless devices that are physically close enough to exchange information. Construction of an ad-hoc network is quick with wireless devices joining and leaving the network as they enter and leave the proximity of the remaining wireless devices. An ad-hoc network also may include one or more access points, that is, stationary wireless devices operating as a stand-alone server or as gateway connections to other networks.

[0005] In the future, the Bluetooth Standard will likely support the interconnection of multiple piconets to form a multi-hop ad-hoc network, or scatternet. In a scatternet, a connecting device forwards traffic between different piconets. The connecting device may serve as a master device in one piconet, but as a slave device or a master device in another piconet. Thus, the connecting devices join the piconets that comprise a scatternet by adapting the timing and hop sequence to the respective piconet and possibly changing the roles that they serve from a master device to a slave device.

[0006] A Bluetooth device includes, but is not limited to, a mobile telephone, personal or laptop computer, radio-frequency identification tag, and personal electronic device such as a personal digital assistant (PDA), pager, or portable-computing device. Each Bluetooth device

includes application and operating system programs designed to find other Bluetooth devices as they enter and leave the communication range of the network. The requesting Bluetooth device in a client role and the responding Bluetooth device in a server role establish a proximity link between the two devices. The requesting and responding Bluetooth device use the proximity link and a service discovery protocol to discover the services offered by the other Bluetooth device and how to connect to those services.

[0007] A public key infrastructure (PKI) is a system of digital certificates, certificate authorities (CAs), and other registration authorities that verify and authenticate the validity of each party involved in an Internet transaction. A digital certificate is an attachment to an electronic message typically to verify that a user sending a message is who they claim to be, and to provide the receiver with the means to encode a reply. An individual wishing to send an encrypted message applies for a digital certificate from a CA. The CA issues a signed digital certificate containing the applicant's public key and a variety of other identification data. The CA makes its own public key readily available through print publicity or perhaps on the Internet. The recipient of an encrypted message uses the CA's public key to verify the digital certificate attached to the message, verify that it was issued by the CA, and obtain the sender's public key and identification information held within the certificate. With this information, the recipient can send an encrypted reply. The most widely used standard for digital certificates is X.509.

[0008] Cryptography is the art of protecting information by transforming (i.e., encrypting) the information into an unreadable format, called cipher text. Only someone who possesses a secret key can decipher (i.e., decrypt) the cipher text into plain text. Symmetric-key

systems and public-key systems are broad classifications of cryptography systems. A symmetric-key system (e.g., the Data Encryption Standard (DES)) is an encryption system in which the sender and receiver of a message share a single, common key that is used to encrypt and decrypt the message. A public-key system (e.g., Pretty Good Privacy (PGP)) uses two keys, a public key known to everyone and a private or secret key known only to the recipient of the message. When John wants to send a secure message to Jane, he uses Jane's public key to encrypt the message. Jane then uses her private key to decrypt the message. Symmetric-key systems are simpler and faster than public-key systems, but their main drawback is that the two parties must somehow exchange the key in a secure way. To avoid this drawback, public-key systems distribute the public key in a non-secure way and never transmit the private key.

[0009] The problem of secure communication and authentication in ad-hoc wireless networks has been addressed in a paper titled *Talking to Strangers: Authentication in Ad-Hoc Wireless Networks* by Balfanz et al. The authors present a solution that provides secure authentication using almost any established public-key-based key exchange protocol, as well as inexpensive hash-based alternatives. The solution allows devices to exchange a limited amount of public information over a privileged side channel, and then allows the devices to complete an authenticated key exchange protocol over the wireless link. The solution does not require a PKI, is secure against passive attacks on the privileged side channel and all attacks on the wireless link, and directly captures the user's intention to communicate with a particular previously unknown device that is within their physical proximity.

[0010] For wireless devices that communicate in a peer-to-peer ad-hoc network, prior art middleware facilitates inter-application communication by hiding peer-discovery, network formation, application and service discovery, as well as automatic application launching, behind an easy-to-use coherent application programming interface (API). However, since no trusted, accessible, third party based solution is available, establishing secure communication and authentication is difficult for the prior art middleware.

[0011] Thus, there is a need for a system and method for providing secure communication between selected applications in wireless ad-hoc network devices that rely upon middleware to facilitate inter-application communication. The system and method will provide the means to implement a security API for application-level access to other security services based on the generated peer-to-peer security associations. The system and method do not require a highly available server or PKI and improve establishment of security by relying on a user to enter a password. The present invention addresses this need.

SUMMARY OF THE INVENTION

[0012] A computer system, method, and computer program product for controlling data communication in an ad-hoc network that connects a wireless device and a nearby wireless device. The method stores an application directory, determines a priority for each entry in the application directory, identifies a selected entry based on the priority, and examines the attributes and security parameters associated with the selected entry. When the security parameters indicate to use a secure connection, the method establishes a security association to support the data communication by querying a database for an existing security association that will satisfy

the security parameters. When the query is successful, the method reuses the existing security association. When the query is unsuccessful, the method creates a new security association by establishing a privileged side channel to the nearby wireless device, negotiating the new security association over the privileged side channel, and storing the new security association.

[0013] The attributes include a device identifier, a role, and control parameters such as an application state and at least one user-defined application setting. The security parameters include an information security objective (e.g., maintaining confidentiality, ensuring integrity, authenticating a party, and protecting against replay or reuse), a cryptography method for attaining the information security objective (e.g., a signature verification service, and an encryption algorithm), and a level of security. In one embodiment, a bit-string includes the security parameters, a value of the bit-string representing each of the security parameters.

[0014] In one embodiment, to reconnect to a secure connection the method stores a security association between the wireless device and the nearby wireless device when the nearby wireless device enters the ad-hoc network for a first encounter. The method stores a copy of the security association so that when the first encounter terminates, the method can remove the security association and retain the copy. When the nearby wireless device enters the ad-hoc network for a second encounter, the method establishes a secure connection to the nearby device based on the copy of the security association. In another embodiment, the method establishes the secure connection by searching a connection log to locate the copy of the security association. In another embodiment, storage of the connection log is on a long-term storage device. In another

embodiment, a user operates a graphical user interface to locate the copy of the security association.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] The accompanying figures best illustrate the details of the system and method for providing secure communication between selected applications in wireless ad-hoc network devices that rely upon middleware to facilitate inter-application communication. Like reference numbers and designations in these figures refer to like elements.

[0016] Figure **1** is a network diagram that illustrates the interaction of the devices that comprise a mobile ad-hoc communications network, in accordance with one embodiment of the present invention.

[0017] Figure **2A** is a block diagram that illustrates the hardware and software components comprising server **110** shown in Figure **1**, in accordance with one embodiment of the present invention.

[0018] Figure **2B** is a block diagram that illustrates the hardware and software components comprising terminal **120** shown in Figure **1**, in accordance with one embodiment of the present invention.

[0019] Figure **3A** and Figure **3B** are flow diagrams of an embodiment of a process for launching and controlling secure and non-secure application programs in a mobile ad-hoc communications network.

[0020] Figure **3C** is a flow diagram of an embodiment of a process for reconnecting a secure application program in a mobile ad-hoc communications network.

[0021] Figure **4** illustrates is a diagram of a window depicting an embodiment of a graphical user interface for reconnecting a secure application program in a mobile ad-hoc communications network.

DETAILED DESCRIPTION OF THE INVENTION

[0022] Figure **1** is a network diagram that illustrates the interaction of the devices that comprise a mobile ad-hoc communications network, in accordance with one embodiment of the present invention. In one embodiment, the mobile ad-hoc communications network is a Bluetooth piconet that includes one master device and up to seven active slave devices. As shown in Figure **1**, piconet **100** includes server **110** and five instances of terminal **120**. Server **110** maintains the network clock and is the communication manager for each instance of terminal **120**. Server **110** typically initiates an exchange of data with an instance of terminal **120**. Two instances of terminal **120** typically communicate through the server **110** however, if two instances of terminal **120** communicate directly, one instance will assume the role of server, or master, and the other instance will assume the role of client, or slave.

[0023] Each device in the mobile ad-hoc communications network will either assume the role of a terminal device or a server device. A terminal device is a consumer of services that a single user operates. A terminal device includes devices such as a mobile phone or PDA. A server is typically a stationary device and only produces services. A server device creates a hotspot around them for using their services. "Hotspot" refers to the radio coverage area

provided by the server device for detecting devices and discovering services offered by the applications hosted in the server. If the server device is not stationary, one of the terminal devices in the network will assume the role of application directory server and perform device detection and service discovery functions for the remaining terminal devices in the network. The disclosed invention introduces two roles among such terminal devices, application directory servers and terminals, where application directory servers serve terminals in device detection and service discovery. If stationary servers with hotspots exist, servers typically act as application directory servers. However, device detection and service discovery is possible without such a stationary server because one of the terminals will assume the application directory server duties.

[0024] The disclosed invention assigns an identifier to each application placed under control. In one embodiment, the identifier is a non-unique identifier that abstractly identifies the application. In another embodiment, the identifier specifies a function that the application performs. In another embodiment, the identifier specifies a communication protocol that the application uses to communicate. Thus, the identifier may indicate that several occurrences of an application each occurrence authored in a different computer language, or targeted to run on a different hardware platform or fulfill a different application role may be considered to be the same because they can interoperate and fulfill the same function. However, in yet another embodiment, the identifier is a unique identifier that identifies the application.

[0025] Figure **2A** is a block diagram that illustrates the hardware and software components comprising server **110** shown in Figure **1**, in accordance with one embodiment of the present invention. Server **110** is a general-purpose wireless device. Bus **200** is a

communication medium that connects keypad **201**, display **202**, central processing unit (CPU) **203**, and radio frequency (RF) adapter **204** to memory **210**. RF adapter **204** connects via a wireless link to terminal **120** and is the mechanism that facilitates network traffic between server **110** and terminal **120**.

[0026] CPU **203** performs the methods of the disclosed invention by executing the sequences of operational instructions that comprise each computer program resident in, or operative on, memory **210**. Memory **210** includes operating system software **211**, application programs **212**, and middleware software **220**. Operating system software **211** controls keypad **201**, display **202**, RF adapter **204**, and the management of memory **210**. Application programs **212** control the interactions between a user and server **110** including a proximity security initialization program. Middleware software **220** includes an application program interface (API) **221**, application directory **230**, security association database **240**, and connection log **245**. API **221** assists an application program running on server **110** to find and communicate with a counterpart application running on terminal **120**. Application directory **230** tracks, for each application that is resident in each device in piconet **100**, a reference to the device storing the application, an identifier for the application, the role that the application performs, and the security parameters that define the required policy configuration attributes and security services. In one embodiment, the reference to the device storing the application is the MAC address of the device. Security association database **240** stores all recent, pair-wise and group associations established by the proximity security initialization program. Connection log **245** stores recent connections to server **110**, such as general packet radio service (GPRS), Bluetooth, or wireless local area network (WLAN) connection. In one embodiment, middleware software **220**

integrates the storage of any combination of application directory **230**, security association database **240**, and connection log **245**.

[0027] Figure **2B** is a block diagram that illustrates the hardware and software components comprising terminal **120** shown in Figure **1**, in accordance with one embodiment of the present invention. Terminal **120** is a general-purpose wireless device. Bus **250** is a communication medium that connects keypad **251**, display **252**, CPU **253**, and RF adapter **254** to memory **260**. RF adapter **254** connects via a wireless link to server **110** or another terminal **120** and is the mechanism that facilitates network traffic between server **110** and terminal **120**.

[0028] CPU **253** performs the methods of the disclosed invention by executing the sequences of operational instructions that comprise each computer program resident in, or operative on, memory **260**. Memory **260** includes operating system software **261**, application programs **262**, and middleware software **270**. Operating system software **261** controls keypad **251**, display **252**, RF adapter **254**, and the management of memory **260**. Application programs **262** control the interactions between a user and terminal **120** including a proximity security initialization program. Middleware software **270** includes an application program interface (API) **271**, application directory **280**, security association database **290**, and connection log **295**. API **271** assists an application program running on server **110** to find and communicate with a counterpart application running on terminal **120**. Application directory **280** tracks, for each application that is resident in each device in piconet **100**, a reference to the device storing the application, an identifier for the application, the role that the application performs, and the security parameters that define the required policy configuration attributes and security services.

In one embodiment, the reference to the device storing the application is the MAC address of the device. Security association database **290** stores all recent, pair-wise and group associations established by the proximity security initialization program. Connection log **295** stores recent connections to terminal **120**, such as general packet radio service (GPRS), Bluetooth, or wireless local area network (WLAN) connection. In one embodiment, middleware software **270** integrates the storage of any combination of application directory **280**, security association database **290**, and connection log **295**.

[0029] In one embodiment, the configuration of memory **210** and memory **260** is identical. In another embodiment, the configuration of memory **210** and memory **260** only includes the software necessary to perform the essential tasks of server **110** and terminal **120**, respectively. For example, if terminal **120** needs to receive a general inquiry access code, but does not need to send a general inquiry access code message, only the software that receives this message will reside in memory **260**.

[0030] In the disclosed invention, the distributed application directory stored in the middleware software is a database that makes it possible for a device to know something of the requirements and wishes of peer devices to which it connects. The database also contains information of local applications and their requirements. The information includes security parameters, as well as priority information, indicating importance of the application set by the user. The distributed application directory, or database, stores these security parameters and the middleware software enforces these security parameters. In one embodiment, these security parameters are stored as a bit-string where the bits allow the user to enable application-level

access control for each entry in the application. Thus, the user may set the security parameters to indicate that a specific application requires that a specific security association is present before communicating with a complementary application running on another device.

[0031] As shown in Figure **2A** and Figure **2B**, the security parameters are a bit-string in which the first four bits identify a communication security type, the next two bits identify a required security API service, and the last two bits identify a level of security. The communication security type identifies the information security objective sought. Information security objectives include keeping information private or confidential, ensuring the integrity of the information, authenticating the identity of the parties to the communication, protecting against replay or reuse of the information, and the like. The specified security API service identifies cryptography methods for required application by obtaining the information security objective. The cryptography method includes signature services, encryption algorithms, and the like. The level of security determines the algorithm as well as the way information is collected. For example, higher-level security may require the use of certain location-limited channels when the security context is established. Other parameters that may be affected by the level of security are the validity period of the established security context or the validity of third-party information. For example, in low-level security formation group keys may well be used for channel protection in a way where pair-wise security establishment may not be needed between every possible pair in the group.

[0032] Middleware software **220** and **270** stores in security association database **240** or **290** all recent, pair-wise and group security associations established by a proximity security

initialization program. However, middleware software **220** and **270** only establishes the security associations between devices, not applications. Middleware software **220** and **270** is also responsible for purging records based on validity period settings and use order (e.g., when the database fills up, purging the oldest and least used associations). For example, if device D and peer P are each running application X, middleware software **220** or **270** sends a query to security association database **240** or **290** for an existing and valid security association between D and P. If security association A exists and satisfies the security parameters associated with X, security association A is used and no other security association between D and P is needed.

[0033] According to one embodiment, a security association includes fields for identifying the peer device and security parameters. The fields for identifying the peer device may specify the local device identity as seen by the peer device (i.e., external) or may specify the identity of the peer device as seen by the local device (i.e., internal). The security parameters, in addition to the security parameters shown in Figure **2A** and Figure **2B**, include a cryptographic digest (e.g., a thumbprint of a certificate), a public key pair, secret keys of a peer device, and a possible lifetime of key material.

[0034] Middleware software **220** and **270** may also prioritize both secure and non-secure applications. To minimize application congestion on the mobile terminal, the prioritization is concerned with runnable, automatically launching applications in the local network. The related application titled "Application Control in Peer-to-Peer Ad-Hoc Communication Networks" describes a system and method for launching and controlling non-secure application programs resident in wireless devices in a spontaneous and instant (ad-hoc) communications network. One

aspect of that system and method chooses the highest priority application and automatically launches the application if the application is "runnable" and the appropriate user-defined flags are set. The system and method disclosed herein addresses the task of prioritizing secure and non-secure, runnable applications that can be automatically launched. Thus, the system and method disclosed herein minimizes "application congestion" on a mobile terminal that includes secure and non-secure applications.

[0035] Middleware software **220** and **270** facilitates inter-application communication by hiding peer discovery, network formation, application and service discovery as well as automatic application launching. When two mobile devices meet, the devices exchange (i.e., distribute) application directory data that describes the applications and peer devices on the network. Using the application directory data, the mobile devices can launch and control application programs resident in wireless devices in a mobile ad-hoc communications network. However, communication in a secure manner requires the integration of the application directory and proximity security initialization software. For each application and terminal in the application directory, the user may set a requirement to use a secure channel, as well as additional information (e.g., the user may set a requirement that the application requires a digital signature facility from a cryptography API). Before launching an application that requires a secure channel, two proximate mobile devices first determine whether an existing security association will support the secure channel communication. If a security association already exists, the matching applications will launch and utilize the security association. If a security association does not exist, both devices will launch proximity security initialization software to use a location-limited (i.e., proximity) side channel, such as an infrared data association (IrDA) port, to

authenticate the devices and negotiate a security association. When the negotiation is complete, the matching applications launch and utilize the negotiated security association.

[0036] Figure **3A** and Figure **3B** are flow diagrams of an embodiment of a process for launching and controlling secure and non-secure application programs in a mobile ad-hoc communications network. The process in Figure **3A** begins when a mobile device (device A) waits for a connection request from the network (step **302**). A proximate mobile device (device B) enters the network and sends a connection request to device A (step **304**). Device A and device B establish a connection (step **306**) and exchange a list of applications and attributes (step **308**). If local application state parameters have changed during the connection (step **310**), device A and device B mutually update their list of applications and attributes (step **312**) and the requesting one of the devices select an application from the list based on a predefined ordering (step **314**). If local application state parameters have not changed during the connection (step **310**), the requesting one of the devices select an application from the list based on a predefined ordering (step **314**).

[0037] If the processing exhausts the list of applications (step **316**), the process returns to waiting for a connection request (step **302**). If the processing selects an application (step **316**) and if the selected application does not require a security association (step **318**), the processing of the selected application continues from step **328** as shown in Figure **3B**. If the selected application requires a security association (step **318**) and if security association database **240** or **290** includes a security context for the selected application (step **320**), the processing of the selected application continues from step **328** as shown in Figure **3B**. If security association

database **240** or **290** does not include a security context for the selected application (step **320**) and if all runnable applications have not executed (step **322**), the process defers the priority of the selected application (step **326**) and continues from step **314** as shown in Figure **3A**. If security association database **240** or **290** does not include a security context for the selected application (step **320**) and if all runnable applications have executed (step **322**), the process negotiates a security association for the deferred secure applications (step **324**) and continues from step **314** as shown in Figure **3A**.

[0038] In another embodiment, rather than defer the priority of the selected application until all runnable applications have executed (step **326**), the process launches the proximity security initialization software on an as needed basis to establish a secure connection for the selected application. In yet another embodiment, the mobile device selects whether to defer the priority or launch the proximity security initialization software based on the computing performance required by the user of the mobile device.

[0039] The process in Figure **3B** begins with examining the attributes associated with the selected application. If the selected application is running in device A (step **328**) and running in device B (step **336**), the process notifies device A and device B of the connection (step **330** and step **338**) and selects the next application from the list (step **314**). If the selected application is running in device A (step **328**), is not running in device B (step **336**), and is startable in device B (step **340**), device B starts the selected application (step **342**) and the process continues from step **318** as shown in Figure **3A**. If the selected application is running in device A (step **328**), is not running in device B (step **336**), is not startable in device B (step **340**), and is missing in device B

(step **344**), then if device B will accept the selected application (step **346**), device A transfers the selected application to device B (step **348**) and the process continues from step **318** as shown in Figure **3A**. If the selected application is not running in device A (step **328**), but is startable in device A (step **332**), the process starts the selected application in device A (step **334**) and continues from step **318** as shown in Figure **3A**. If the selected application is running in device A (step **328**), is not running in device B (step **336**), is not startable in device B (step **340**), and is not missing from device B (step **344**), the process continues from step **322** as shown in Figure **3A**. If the selected application is not running in device A (step **328**) and is not startable in device A (step **332**), the process continues from step **322** as shown in Figure **3A**.

[0040] To negotiate the security association, middleware software **220** or **270** launches proximity security initialization software. The proximity security initialization software enables two devices, such as server **110** and terminal **120**, that have no prior security context to authenticate each other based on some kind of user-initiated physical authentication and in a resulting communication protocol generate a security association. There mutual authentication protocols include each user entering a common password into their device, using a location-limited channel as described in Balfanz, visual or short has mutual verification by the users, and pair-wise biometric identification (i.e., entering biometric data in the peer device that the local device authenticates).

[0041] If a security association between two peer devices does not exist, middleware software **220** or **270** for the first peer device sends a request to establish a security association between the first and the second peer device. The proximity security initialization software

resident in the first device negotiates the security association, including peer identification data, over a location-limited channel. Once negotiation of the security association is complete, the proximity security initialization software offers the location-limited channel for use to middleware software **220** or **270** and the application that requires the security association.

[0042] If the security association between the two peer devices exists, before an application that requires security can launch, the proximity security initialization software enforces the security policy set in application directory **230** or **280**. The proximity security initialization software retrieves the correct security association from middleware **220** or **270**, and configures the necessary security protocols (e.g., Bluetooth pairing, or Transport Layer Security/Internet Protocol Security). The application that requires security launches after establishment of the security services. However, if the security policy for the application requests a security application program interface (API) the proximity security initialization software also configures the required cryptographic services. In another embodiment, when critical security levels are in use, the security associations stored in middleware **220** or **270** may be protected and require user interaction (e.g., entering a password, or providing biometric data).

[0043] In another embodiment, the definitions of the security associations are application specific. This approach requires the proximity security initialization software to negotiate a specific security association for each pair-wise application that needs security according to the associated security policy. Also, the proximity security initialization software and the communication infrastructure provide support for multiplexing several security contexts over the communications channel based on the associations in use.

[0044] Figure **3C** is a flow diagram of an embodiment of a process for reconnecting a secure application program in a mobile ad-hoc communications network. If security association database **240** or **290** does not include a security context for the selected application (step **320**), the process optionally sends a request to middleware software **220** and **270** for data from connection log **245** or **295** (step **350**). The process stores the data supplied in response to the request in a display list (step **352**). The display list provides a user with the ability to browse the display list (step **354**), view detailed data for each entry in the display list (step **356**), and select an entry in the display list (step **358**). If the entry selected is a previously established connection (step **360**), the process continues from step **328** as shown in Figure **3B**. If the entry selected is not a previously established connection (step **360**), the process continues from step **322** as shown in Figure **3A**.

[0045] Connection log **245** and **295**, and security association database **240** and **290**, store in a similar manner initialization parameters that describe a secure application connection. These initialization parameters include fields for identifying the device, and fields for security parameters. The fields for identifying the device include, for example, the local device identity as seen by the peer device, and the peer device identity as seen by the local device. The fields for security parameters include, for example, Cryptographic Digest such as a thumbprint of a certificate, public key of the peer device, secret keys of the peer device, and the possible lifetime of the key material. However, in contrast, connection log **245** and **295** is longer-term storage than security association database **240** and **290**. For example, security association database **240** and **290** may only store the initialization parameters for the duration of the application connection and release the storage of those parameters when the user exits the application. Since

connection log **245** and **295** retains a copy of those initialization parameters for a longer period of time, the peer device can leave the ad-hoc network and upon returning to the network immediately start the application using the initialization parameters. Thus, the returning peer device need not re-establish a location-limited channel to touch the local device and begin secure communications.

[0046] Figure **4** illustrates is a diagram of a window depicting an embodiment of a graphical user interface for reconnecting a secure application program in a mobile ad-hoc communications network. The graphical user interface shown in Figure **4** is resident in terminal **120** as shown in detail in Figure **2B**. Thus, like reference numbers and designations in Figures **2B** and **4** refer to like elements. However, a reader of this disclosure should understand that the graphical user interface might reside similarly in server **110** as shown in detail in Figure **2A**.

[0047] As shown in Figure **4**, window **400** resides in display **252** of terminal **120**. Window **400** includes the elements comprising the graphical user interface. Bus **250** is a communication medium that connects keypad **251**, display **252**, CPU **253**, RF adapter **254**, and memory **260**. Figure **4** shows display **252** and memory **260** as separate components. In another embodiment, CPU **253** stores window **400** is a display or video memory associated with display **252**. However, in yet another embodiment, CPU **253** stores window **400** in a protected portion of memory **260**.

[0048] Window **400** shown in Figure **4** includes title bar **410**, data header **420**, display list **430**, scroll bar **440**, and index **450**. Title bar **410** identifies the title of window **400** as "Connection Log". Data header **420** identifies the title for the columns comprising each entry in

display list **430**. Display list **430** includes one entry for each entry in connection log **295**. If display list **430** is an empty list, there is no selected entry. If display list **430** is not an empty list, the graphical user interface will always consider one entry to be a selected entry and will display the selected entry in a format that differs visually from the display of the non-selected entries. Figure **4** depicts the selected entry shown in reverse video as the entry in which "Device Name" is "Carol's Phone". Scroll bar **440** is a navigational element that provides an indication of the spatial location of the selected entry in display list **430**. Scroll bar **440** includes an up arrow, a down arrow, a scroll region, and a scroll box in the scroll region. If the selected entry is the first entry in display list **430**, the location of the scroll box is at the top-most portion of the scroll region. If the selected entry is the last entry in display list **430**, the location of the scroll box is at the bottom-most portion of the scroll region. Otherwise, the location of the scroll box within the scroll region is proportional to the quotient of the index of the selected entry in display list **430** and the index of the last entry in display list **430**, where the index of the first entry in display list **430** is one and the index of the last entry in display list **430** is equal to the number of entries in display list **430**. Index **450** is another navigational element that provides an indication of the numerical location of the selected entry in display list **430**. Index **450** includes two numbers separated by a slash. The number to the left of the slash is the index associated with the selected entry. The number to the right of the slash is the index associated with the last entry in display list **430** (i.e., the number of entries in display list **430**).

[0049] Referring again to Figure **4**, CPU **253** executes the appropriate instructions to cause window **400** to appear in display **252** of terminal **120**. CPU **253** sends a request to middleware software **270** via bus **250** for data from connection log **295**. CPU **253** executes the

appropriate instructions to store the data received in response to the request in display list **430**. A user of terminal **120** may browse display list **430** shown in display **252**. The user may also view detailed data for each entry in display list **430**. The detailed data includes a name for the device (e.g., "Carol's Phone), a timestamp associated with the connection (e.g., October 3, 2003, 20 hours, and 42 seconds), an indication of the presence of security parameters, and a channel type for the connection (e.g., RFID). The user may also operate an input device such as keypad **251** to change the entry in display list **430** that is the selected entry, and to choose to reconnect to the secure application program associated with the selected entry. In another embodiment, the user may operate keypad **251** to display additional detailed data associated with the selected entry.

[0050] Although the disclosed embodiments describe a fully functioning system and method for launching and controlling secure and non-secure application programs in wireless devices in a mobile ad-hoc communications network, the reader should understand that other equivalent embodiments exist. Since numerous modifications and variations will occur to those who review this disclosure, the system and method for launching and controlling secure and non-secure application programs resident in wireless devices in a mobile ad-hoc communications network is not limited to the exact construction and operation illustrated and disclosed. Furthermore, the disclosed invention may be distributed in the form of a computer readable medium of instructions including recordable media such as a removable disc, a hard disk drive, random access memory, flash memory, and read-only memory, as well as transmission media such as a digital or analog communication link. Accordingly, this disclosure intends all suitable modifications and equivalents to fall within the scope of the claims.